UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/692,323	10/23/2003	Mark A. Alcazar	MS1-1800US	8608

22801 7590 11/23/2009

LEE & HAYES, PLLC
601 W. RIVERSIDE AVENUE
SUITE 1400
SPOKANE, WA 99201

EXAMINER
CHEN, QING

ART UNIT	PAPER NUMBER
2191	

NOTIFICATION DATE	DELIVERY MODE
11/23/2009	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

lhptoms@leehayes.com

Office Action Summary	Application No.	Applicant(s)	
	10/692,323	ALCAZAR ET AL.	
	Examiner	Art Unit	
	Qing Chen	2191	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *02 July 2009*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *27-46* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *27-46* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. This Office action is in response to the amendment filed on July 2, 2009.

2. **Claims 27-46** are pending.

3. **Claims 27, 45, and 46** have been amended.

4. **Claims 1-26** have been canceled.

5. Applicant's amendments to the claims fail to fully address the objections to Claims 27-46 due to improper antecedent basis. Accordingly, these objections are maintained and further explained hereinafter.

Response to Amendment

Claim Objections

6. **Claims 27-46** are objected to because of the following informalities:

- **Claims 27, 45, and 46** recite the limitation "the resource." Applicant is advised to change this limitation to read "the requested resource" for the purpose of providing it with proper explicit antecedent basis.
- **Claims 28-44** depend on Claim 27 and, therefore, suffer the same deficiency as Claim 27.

Appropriate correction is required.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. **Claims 27-31, 34, 36-39, and 42-44** are rejected under 35 U.S.C. 103(a) as being unpatentable over **US 7,062,567 (hereinafter "Benitez")** in view of **US 6,442,754 (hereinafter "Curtis")**.

As per **Claim 27**, Benitez discloses:

- invoke a deployment manifest to obtain manifest metadata about an application for the purpose of installing the application on a client computing system *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.")*;

- receive the manifest metadata about the application *(see Column 14: 27-32, "The Application Stream Builder creates the Stream App Install Block 405 used to set up a client system for Streaming Application Delivery and Execution and it also creates the set of Application File Pages 406 sent to satisfy client requests by the Application Server 107.")*;

- determine whether the application is authorized for installation on the client computing system *(see Column 13: 57-67, "The License Server 106 checks the Subscription 101 and License 102 Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.")*; and

- enable the application to be installed on the client computing system, wherein during the enabled installation, the application is available for use while being installed *(see Column 12: 6-21, "Client Application Installer 305—This component is invoked when the application needs to be installed. The Client Application Installer 305 sends a specific request to the Application Server 107 for getting the Stream App Install Block 301 for the particular application that needs to be installed."; Column 15: 58-63, "The streaming file system allows applications to be run immediately by retrieving application file contents from the server as they are needed, not as the application is installed. This removes the download cost penalty of doing local installations of the application.")*, wherein installation on the client computing system comprises:

- receiving a request from the client computing system for a resource *(see Column 8: 57-61, "Once the client 113 obtains an "Access token" to run an application, it connects to the Application Server 107 and presents to it the "Access token" along with the request for the application bits.")*;

- determining if the requested resource is stored locally on the client computing system *(see Column 10: 37-42, "The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212. If it does not have the bits cached [determining if the requested resource is stored locally on the client computing system], it gets them from the Application Server 107 through the network interface.")*;

- if the requested resource is not stored locally on the client computing system, determining if the requested resource is an on demand resource or an online resource *(see Column 10: 57-61, "The Client Cache Manager 207 will send those bits from the cache if they exist there or forward the request to the Application Server 107 through the network interface to get the appropriate bits.")*; [Examiner's Remarks: Note that the Client Cache Manager forwards to requests to the Application Server to get the appropriate application bits by determining that the requested application bits are online resources to be streamed.]

- if the requested resource is an on demand resource, caching the requested resource in an application store *(see Figure 2: 210; Column 11: 11-18, "Client File Spoofer 211--Certain files [on demand resource] on the client need to be installed at specific locations on the client system. To be able to stream these files from the Application Server 107, the Client Spoofer 211 intercepts all requests to these files made by a running application and redirects them to the Client Streaming File System 212 so that the bits can be streamed from the Application Server 107."; Column 18: 38-40, "Code or data retrieved from the server 802 will be placed in the cache in case it is used again.")*;

- if the requested resource is an online resource, caching the requested resource in a transient cache and copying the requested resource into the application store *(see Figure 2: 206; Column 10: 27-31, "Client Cache Manager 207--This component caches the application bits [online resource] received from the Application Server 107 so that next time a request is made to the same bits, the request can be served by the cache instead of having to go to the Application Server 107."; Column 17: 53-57, "The client software first checks its cache 611, 620 in*

nonvolatile storage for the requested code or data. If it is found there, the code or data are copied from the cache in nonvolatile storage 620 to volatile memory 619."); and

- installing the application when a number of resources stored in the application store reaches a threshold number *(see Column 10: 32-40, "The Client Cache Manager 207 has a limited amount of space on the disk of the client machine that it uses for the cache. When the space is fully occupied, the Client Cache Manager 207 uses a policy to replace existing portions of the cache. This policy can be something like LRU, FIFO, random etc. The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212." and 53-57, "These requests lead to page faults in the operating system and the page faults are handled by the Client Streaming File System 212 that in turn asks the Client Cache Manager 207 for the appropriate bits.")*. [Examiner's Remarks: Note that the cache stores the application bits that are provided to the Client Streaming File System when requested (installing the application). When the cache is fully occupied (reaches a threshold number), various cache algorithms may be used to replace existing portions of the cache so that other requested application bits can be provided to the Client Streaming File System when requested.]

However, Benitez does not disclose:

- issue a query of an install state of the client computing system to determine whether a platform necessary to the application is present on the client computing system; and

- receive the install state of the necessary platform present on the client computing system.

Curtis discloses:

- issue a query of an install state of a client computing system to determine whether a platform necessary to an application is present on the client computing system *(see Column 3: 61-67, "The program then executes the operating system command to determine whether the dependent components indicated in the dependency objects are installed in the computer.")*; and

- receive the install state of the necessary platform present on the client computing system *(see Column 3: 61-67, "An indication is made as to the dependent components that are not installed after determining that dependent components are not installed.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Curtis into the teaching of Benitez to modify Benitez's invention to include issue a query of an install state of the client computing system to determine whether a platform necessary to the application is present on the client computing system; and receive the install state of the necessary platform present on the client computing system. The modification would be obvious because one of ordinary skill in the art would be motivated to check whether any required dependent components are installed in the client system when installing a program *(see Curtis – Column 3: 44-48)*.

As per **Claim 28**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the manifest metadata includes information sufficient to describe the application *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of*

the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.").

As per **Claim 29**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API receives a parameter that identifies the application *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.").*

As per **Claim 30**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API invokes a deployed application identity to obtain the manifest metadata about the application *(see Column 9: 60-67, "e. Application File*

Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.").

As per **Claim 31**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API invokes both a deployment manifest and a deployed application identity to obtain the manifest metadata about the application *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.").*

As per **Claim 34**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API will generate a set of authorization parameters for an authorized application *(see Column 13: 57-67, "The License Server 106 checks the Subscription 101 and License 102 Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.")*.

As per **Claim 36**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the platform comprises one or more software modules upon which the application depends that are not part of the application *(see Column 7: 7-22, "... there are certain shared library files, e.g., "foo.dll", that need to be installed on the local file system, e.g., "c:\winnt\system32\foo.dll", for the application to execute.")*.

As per **Claim 37**, the rejection of **Claim 36** is incorporated; and Benitez further discloses:

- wherein the platform further comprises one or more software modules that cannot be installed as part of the installation of the application *(see Column 7: 7-22, "For the previous example, the spoof database would contain an entry saying that "c:\winnt\system32\foo.dll" is mapped to "z:\word\winnt\system32\foo.dll" where "z:" implies that this file is accessed by the Client Streaming File System. The Client Spoofer will then redirect all accesses to "c:\winnt\system32\foo.dll" to "z:\word\winnt\system32\foo.dll". In this manner, the client system*

gets the effect of the file being on the local machine whereas in reality the file is streamed from the server.").

As per **Claim 38**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the platform is identified in an application manifest associated with the application *(see Column 7: 7-9, "The invention employs a Client Streaming File System that is used to manage specific application-related file accesses during the execution of an application.")*.

As per **Claim 39**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API includes verifying a version associated with the platform *(see Column 17: 23-33, "If certain code segments need to be updated, then the code segment listing in the application root directory is simply changed and the new code segment subdirectory added. This results in the new and correct code segment subdirectory being read when it is referenced.")*.

As per **Claim 42**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the determination of the authorization comprises determining whether the installation of the application violates a license associated with the application *(see Column 13:*

57-67, "The License Server 106 checks the Subscription 101 and License 102 Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.").

As per **Claim 43**, the rejection of **Claim 27** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API includes determining if a version of the application already exists on the client computing system *(see Column 21: 17-28, "When retrieving an old file that hasn't changed, it will find the old file identifier, which can be used for the existing files in the cache. In this way, files that do not change can be reused from the cache without downloading them again.").*

As per **Claim 44**, the rejection of **Claim 43** is incorporated; and Benitez further discloses:

- wherein the computer-implemented API includes downloading at least one resource associated with the application if the application does not exist on the client computing system *(see Column 14: 15-26, "The Client Application Installer 607 opens and reads that file (which engages the Client Streaming File System) and updates the Client system appropriately, including setting up the spoof database, downloading certain needed non-application-specific files, modifying the registry file, and optionally providing a list of applications pages to be prefetched to warm up the Client Stream Cache 611 with respect to the application.").*

9. **Claims 32 and 33** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis** as applied to Claim 27 above, and further in view of **US 6,496,979 (hereinafter "Chen")**.

As per **Claim 32**, the rejection of **Claim 27** is incorporated; however, Benitez and Curtis do not disclose:

- wherein the computer-implemented API will abort the installation of the application if the platform is not present.

Chen discloses:

- wherein a computer-implemented API will abort an installation of an application if a platform is not present *(see Column 11: 43-51, "... the installer module 99 can provide an indication to the user that the setup package file contains files that were compiled for a mobile device different than the current one and let the user continue or cancel the installation.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chen into the teaching of Benitez to modify Benitez's invention to include wherein the computer-implemented API will abort the installation of the application if the platform is not present. The modification would be obvious because one of ordinary skill in the art would be motivated to resolve any installation problems before the application setup program is in its final product state *(see Chen – Column 2: 21-28)*.

As per **Claim 33**, the rejection of **Claim 32** is incorporated; however, Benitez and Curtis do not disclose:

- wherein the computer-implemented API will return error information in conjunction with aborting the installation of the application.

Chen discloses:

- wherein a computer-implemented API will return error information in conjunction with aborting an installation of an application *(see Column 10: 55-61, "... determines that the map viewer is not installed and displays an error message ...")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chen into the teaching of Benitez to modify Benitez's invention to include wherein the computer-implemented API will return error information in conjunction with aborting the installation of the application. The modification would be obvious because one of ordinary skill in the art would be motivated to provide debugging information as to why the application cannot be installed.

10. **Claim 35** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis** as applied to Claim 34 above, and further in view of **US 6,931,546 (hereinafter "Kouznetsov")** and **US 2002/0104015 (hereinafter "Barzilai")**.

As per **Claim 35**, the rejection of **Claim 34** is incorporated; and Benitez further discloses:

- wherein the set of authorization parameters comprise at least license keys *(see Column 13: 57-67, "The License Server 106 checks the Subscription 101 and License 102*

Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.").

However, Benitez and Curtis do not disclose:

- wherein the set of authorization parameters comprise at least permission grants and privacy policy guarantees.

Kouznetsov discloses:

- wherein a set of authorization parameters comprises at least permission grants *(see Column 4: 35-38, "The agent includes methods for authenticating any received requests and will only forward a request to the privileged process upon determining that the requesting application has sufficient trust.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Kouznetsov into the teaching of Benitez to modify Benitez's invention to include wherein the set of authorization parameters comprises at least permission grants. The modification would be obvious because one of ordinary skill in the art would be motivated to provide additional means of access authorization for the software programs to prevent any unauthorized access by setting proper permissions.

Barzilai discloses:

- wherein a set of authorization parameters comprises at least privacy policy guarantees *(see Paragraph [0072], "An application request handler 50 receives and processes information requests from application 36 and returns information that is provided by personal information engine 44, to the extend permitted by privacy policies.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Barzilai into the teaching of Benitez to modify Benitez's invention to include wherein the set of authorization parameters comprises at least privacy policy guarantees. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a secured operating environment for the software programs by disclosing information about the use and protection of private data collected by the software programs.

11. **Claim 40** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis** as applied to Claim 27 above, and further in view of **Kouznetsov**.

As per **Claim 40**, the rejection of **Claim 27** is incorporated; however, Benitez and Curtis do not disclose:

- wherein the determination of the authorization comprises determining whether the installation of the application exceeds a trust level associated with a source of the application.

Kouznetsov discloses:

- wherein a determination of an authorization comprises determining whether an installation of an application exceeds a trust level associated with a source of the application *(see Column 4: 35-38, "The agent includes methods for authenticating any received requests and will only forward a request to the privileged process upon determining that the requesting application has sufficient trust.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Kouznetsov into the teaching of Benitez to modify Benitez's invention to include wherein the determination of the authorization comprises determining whether the installation of the application exceeds a trust level associated with a source of the application. The modification would be obvious because one of ordinary skill in the art would be motivated to use a trust level to guard access to privileged processes *(see Kouznetsov – Column 3: 43-44)*.

12. **Claim 41** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis** as applied to Claim 27 above, and further in view of **Barzilai**.

As per **Claim 41**, the rejection of **Claim 27** is incorporated; however, Benitez and Curtis do not disclose:

- wherein the determination of the authorization comprises determining whether the installation of the application violates a privacy policy associated with the client computing system.

Barzilai discloses:

- wherein a determination of an authorization comprises determining whether an installation of an application violates a privacy policy associated with a client computing system *(see Paragraph [0072], "An application request handler 50 receives and processes information requests from application 36 and returns information that is provided by personal information engine 44, to the extend permitted by privacy policies.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Barzilai into the teaching of Benitez to modify Benitez's invention to include wherein the determination of the authorization comprises determining whether the installation of the application violates a privacy policy associated with the client computing system. The modification would be obvious because one of ordinary skill in the art would be motivated to use a privacy policy to protect private information *(see Barzilai – Paragraph [0004])*.

13. **Claim 45** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis** and **Chen**.

As per **Claim 45**, Benitez discloses:

- invoke a deployment manifest to obtain manifest metadata about an application for the purpose of installing the application on a client computing system *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f. Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.")*;

- receive the manifest metadata about the application *(see Column 14: 27-32, "The Application Stream Builder creates the Stream App Install Block 405 used to set up a client system for Streaming Application Delivery and Execution and it also creates the set of Application File Pages 406 sent to satisfy client requests by the Application Server 107.")*;

- determine whether the application is authorized for installation on the client computing system *(see Column 13: 57-67, "The License Server 106 checks the Subscription 101 and License 102 Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.")*; and

- enable the application to be installed on the client computing system, wherein during the enabled installation, the application is available for use while being installed *(see Column 12: 6-21, "Client Application Installer 305—This component is invoked when the application needs to be installed. The Client Application Installer 305 sends a specific request to the Application Server 107 for getting the Stream App Install Block 301 for the particular application that needs to be installed."; Column 15: 58-63, "The streaming file system allows applications to be run immediately by retrieving application file contents from the server as they are needed, not as the application is installed. This removes the download cost penalty of doing local installations of the application.")*, wherein installation on the client computing system comprises:

- receiving a request from the client computing system for a resource *(see Column 8: 57-61, "Once the client 113 obtains an "Access token" to run an application, it connects to the Application Server 107 and presents to it the "Access token" along with the request for the application bits.")*;

- determining if the requested resource is stored locally on the client computing system *(see Column 10: 37-42, "The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212. If it does not have the bits cached [determining if the requested resource is stored locally on the client computing system], it gets them from the Application Server 107 through the network interface.")*;

- if the requested resource is not stored locally on the client computing system, determining if the requested resource is an on demand resource or an online resource *(see Column 10: 57-61, "The Client Cache Manager 207 will send those bits from the cache if they exist there or forward the request to the Application Server 107 through the network interface to get the appropriate bits.")*; [Examiner's Remarks: Note that the Client Cache Manager forwards to requests to the Application Server to get the appropriate application bits by determining that the requested application bits are online resources to be streamed.]

- if the requested resource is an on demand resource, caching the requested resource in an application store *(see Figure 2: 210; Column 11: 11-18, "Client File Spoofer 211--Certain files [on demand resource] on the client need to be installed at specific locations on the client system. To be able to stream these files from the Application Server 107, the Client Spoofer 211 intercepts all requests to these files made by a running application and redirects them to the Client Streaming File System 212 so that the bits can be streamed from the Application Server 107."; Column 18: 38-40, "Code or data retrieved from the server 802 will be placed in the cache in case it is used again.")*;

- if the requested resource is an online resource, caching the requested resource in a transient cache and copying the requested resource into the application store *(see Figure 2: 206;*

Column 10: 27-31, "Client Cache Manager 207--This component caches the application bits [online resource] received from the Application Server 107 so that next time a request is made to the same bits, the request can be served by the cache instead of having to go to the Application Server 107."; Column 17: 53-57, "The client software first checks its cache 611, 620 in nonvolatile storage for the requested code or data. If it is found there, the code or data are copied from the cache in nonvolatile storage 620 to volatile memory 619."); and

- installing the application when a number of resources stored in the application store reaches a threshold number *(see Column 10: 32-40, "The Client Cache Manager 207 has a limited amount of space on the disk of the client machine that it uses for the cache. When the space is fully occupied, the Client Cache Manager 207 uses a policy to replace existing portions of the cache. This policy can be something like LRU, FIFO, random etc. The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212." and 53-57, "These requests lead to page faults in the operating system and the page faults are handled by the Client Streaming File System 212 that in turn asks the Client Cache Manager 207 for the appropriate bits.")*. [Examiner's Remarks: Note that the cache stores the application bits that are provided to the Client Streaming File System when requested (installing the application). When the cache is fully occupied (reaches a threshold number), various cache algorithms may be used to replace existing portions of the cache so that other requested application bits can be provided to the Client Streaming File System when requested.]

However, Benitez does not disclose:

- issue a query of an install state of the client computing system to determine whether a platform necessary to the application is present on the client computing system, wherein the

installation of the application is aborted if the platform is not present and error information is returned in conjunction with aborting the installation of the application; and

- receive the install state of the necessary platform present on the client computing system.

Curtis discloses:

- issue a query of an install state of a client computing system to determine whether a platform necessary to an application is present on the client computing system *(see Column 3: 61-67, "The program then executes the operating system command to determine whether the dependent components indicated in the dependency objects are installed in the computer.")*; and

- receive the install state of the necessary platform present on the client computing system *(see Column 3: 61-67, "An indication is made as to the dependent components that are not installed after determining that dependent components are not installed.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Curtis into the teaching of Benitez to modify Benitez's invention to include issue a query of an install state of the client computing system to determine whether a platform necessary to the application is present on the client computing system; and receive the install state of the necessary platform present on the client computing system. The modification would be obvious because one of ordinary skill in the art would be motivated to check whether any required dependent components are installed in the client system when installing a program *(see Curtis – Column 3: 44-48)*.

Chen discloses:

- wherein an installation of an application is aborted if a platform is not present and error information is returned in conjunction with aborting the installation of the application *(see Column 10: 55-61, "... determines that the map viewer is not installed and displays an error message ..."; Column 11: 43-51, "... the installer module 99 can provide an indication to the user that the setup package file contains files that were compiled for a mobile device different than the current one and let the user continue or cancel the installation.").*

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Chen into the teaching of Benitez to modify Benitez's invention to include wherein the installation of the application is aborted if the platform is not present and error information is returned in conjunction with aborting the installation of the application. The modification would be obvious because one of ordinary skill in the art would be motivated to resolve any installation problems before the application setup program is in its final product state *(see Chen – Column 2: 21-28).*

14. **Claim 46** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Benitez** in view of **Curtis**, **Kouznetsov**, and **Barzilai**.

As per **Claim 46**, Benitez discloses:

- invoke a deployment manifest to obtain manifest metadata about an application for the purpose of installing the application on a client computing system *(see Column 9: 60-67, "e. Application File Pages 111—This is the one of the outputs of the "builder" as explained below and is put on the Application Server 107 so that it can serve the appropriate bits to the client. f.*

Stream App Install Blocks 112—This is the other output of the "builder" and contains the information for successfully installing applications on the client for streaming applications."; Column 14: 15-19, "Whenever the user chooses to install an application, the Client License Manager 608 passes the request to the Client Application Installer 607 along with the name of the Stream App Install Block to be obtained from the Application Server 107.");

- receive the manifest metadata about the application *(see Column 14: 27-32, "The Application Stream Builder creates the Stream App Install Block 405 used to set up a client system for Streaming Application Delivery and Execution and it also creates the set of Application File Pages 406 sent to satisfy client requests by the Application Server 107.")*;

- determine whether the application is authorized for installation on the client computing system, wherein the computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least license keys *(see Column 13: 57-67, "The License Server 106 checks the Subscription 101 and License 102 Databases and, if the user has the right to hold the license at the current time, it sends back an Access Token, which represents the right to use the license.")*; and

- enable the application to be installed on the client computing system, wherein during the enabled installation, the application is available for use while being installed *(see Column 12: 6-21, "Client Application Installer 305—This component is invoked when the application needs to be installed. The Client Application Installer 305 sends a specific request to the Application Server 107 for getting the Stream App Install Block 301 for the particular application that needs to be installed."; Column 15: 58-63, "The streaming file system allows applications to be run immediately by retrieving application file contents from the server as they are needed, not as the*

application is installed. This removes the download cost penalty of doing local installations of the application."), wherein installation on the client computing system comprises:

- receiving a request from the client computing system for a resource *(see Column 8: 57-61, "Once the client 113 obtains an "Access token" to run an application, it connects to the Application Server 107 and presents to it the "Access token" along with the request for the application bits.")*;

- determining if the requested resource is stored locally on the client computing system *(see Column 10: 37-42, "The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212. If it does not have the bits cached [determining if the requested resource is stored locally on the client computing system], it gets them from the Application Server 107 through the network interface.")*;

- if the requested resource is not stored locally on the client computing system, determining if the requested resource is an on demand resource or an online resource *(see Column 10: 57-61, "The Client Cache Manager 207 will send those bits from the cache if they exist there or forward the request to the Application Server 107 through the network interface to get the appropriate bits.")*; [Examiner's Remarks: Note that the Client Cache Manager forwards to requests to the Application Server to get the appropriate application bits by determining that the requested application bits are online resources to be streamed.]

- if the requested resource is an on demand resource, caching the requested resource in an application store *(see Figure 2: 210; Column 11: 11-18, "Client File Spoofer 211--Certain files [on demand resource] on the client need to be installed at specific locations on the client system. To be able to stream these files from the Application Server 107, the Client Spoofer*

211 intercepts all requests to these files made by a running application and redirects them to the Client Streaming File System 212 so that the bits can be streamed from the Application Server 107."; Column 18: 38-40, "Code or data retrieved from the server 802 will be placed in the cache in case it is used again.");

- if the requested resource is an online resource, caching the requested resource in a transient cache and copying the requested resource into the application store *(see Figure 2: 206; Column 10: 27-31, "Client Cache Manager 207--This component caches the application bits [online resource] received from the Application Server 107 so that next time a request is made to the same bits, the request can be served by the cache instead of having to go to the Application Server 107."; Column 17: 53-57, "The client software first checks its cache 611, 620 in nonvolatile storage for the requested code or data. If it is found there, the code or data are copied from the cache in nonvolatile storage 620 to volatile memory 619.")*; and

- installing the application when a number of resources stored in the application store reaches a threshold number *(see Column 10: 32-40, "The Client Cache Manager 207 has a limited amount of space on the disk of the client machine that it uses for the cache. When the space is fully occupied, the Client Cache Manager 207 uses a policy to replace existing portions of the cache. This policy can be something like LRU, FIFO, random etc. The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212." and 53-57, "These requests lead to page faults in the operating system and the page faults are handled by the Client Streaming File System 212 that in turn asks the Client Cache Manager 207 for the appropriate bits.")*. [Examiner's Remarks: Note that the cache stores the application bits that are provided to the Client Streaming File System when requested

(installing the application). When the cache is fully occupied (reaches a threshold number), various cache algorithms may be used to replace existing portions of the cache so that other requested application bits can be provided to the Client Streaming File System when requested.]

However, Benitez does not disclose:

- issue a query of an install state of the client computing system to determine whether a platform necessary to the application is present on the client computing system;

- receive the install state of the necessary platform present on the client computing system; and

- wherein the computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least permission grants and privacy policy guarantees.

Curtis discloses:

- issue a query of an install state of a client computing system to determine whether a platform necessary to an application is present on the client computing system *(see Column 3: 61-67, "The program then executes the operating system command to determine whether the dependent components indicated in the dependency objects are installed in the computer.")*; and

- receive the install state of the necessary platform present on the client computing system *(see Column 3: 61-67, "An indication is made as to the dependent components that are not installed after determining that dependent components are not installed.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Curtis into the teaching of Benitez to modify Benitez's invention to include issue a query of an install state of the client computing system to

determine whether a platform necessary to the application is present on the client computing system; and receive the install state of the necessary platform present on the client computing system. The modification would be obvious because one of ordinary skill in the art would be motivated to check whether any required dependent components are installed in the client system when installing a program *(see Curtis – Column 3: 44-48)*.

Kouznetsov discloses:

- wherein a computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least permission grants *(see Column 4: 35-38, "The agent includes methods for authenticating any received requests and will only forward a request to the privileged process upon determining that the requesting application has sufficient trust.")*.

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Kouznetsov into the teaching of Benitez to modify Benitez's invention to include wherein the computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least permission grants. The modification would be obvious because one of ordinary skill in the art would be motivated to provide additional means of access authorization for the software programs to prevent any unauthorized access by setting proper permissions.

Barzilai discloses:

- wherein a computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least privacy policy guarantees *(see Paragraph [0072], "An application request handler 50 receives and processes information requests from*

application 36 and returns information that is provided by personal information engine 44, to the extend permitted by privacy policies.").

Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teaching of Barzilai into the teaching of Benitez to modify Benitez's invention to include wherein the computer-implemented API will generate a set of authorization parameters for an authorized application comprising at least privacy policy guarantees. The modification would be obvious because one of ordinary skill in the art would be motivated to provide a secured operating environment for the software programs by disclosing information about the use and protection of private data collected by the software programs.

Response to Arguments

15. Applicant's arguments filed on July 2, 2009 have been fully considered, but they are not persuasive.

In the Remarks, Applicant argues:

a) Benitez does not disclose "caching the requested resource into a transient cache and copying the requested resource in the application store" as presently claimed by independent claim 27. Instead, Benitez discloses that when "the space is fully occupied, the Client Cache Manager 207 uses a policy to replace existing portions of the cache. This policy can be something like LRU, FIFO, random etc." (See Benitez, Column 10, lines 27-48). Benitez teaches that the client cache manager requests application bits through the network interface as necessary. Id. When the limited amount of cache is used up, Benitez discloses elimination of

application bits. Id. Benitez is silent as to copying the requested resources cached in the transient cache to the application store.

Examiner's response:

a) Examiner disagrees. With respect to the Applicant's assertion that Benitez is silent as to copying the requested resources cached in the transient cache to the application store, the Examiner respectfully submits that Benitez clearly discloses "copying the requested resource into the application store" *(see Column 17: 53-57, "The client software first checks its cache 611, 620 in nonvolatile storage for the requested code or data. If it is found there, the code or data are copied from the cache in nonvolatile storage 620 to volatile memory 619.").*

Therefore, for at least the reason set forth above, the rejections made under 35 U.S.C. § 103(a) with respect to Claims 27, 45, and 46 are proper and therefore, maintained.

In the Remarks, Applicant argues:

b) Independent claim 27 has also been amended to clarify that the application is installed "when a number of resources stored in the application store reaches a threshold number." This aspect was not previously recited by any claim; accordingly, the Office has not cited any prior art as disclosing, teaching or suggesting this aspect.

Examiner's response:

b) Examiner disagrees. Examiner respectfully submits that Benitez clearly discloses "installing the application when a number of resources stored in the application store reaches a

threshold number" *(see Column 10: 32-40, "The Client Cache Manager 207 has a limited amount of space on the disk of the client machine that it uses for the cache. When the space is fully occupied, the Client Cache Manager 207 uses a policy to replace existing portions of the cache. This policy can be something like LRU, FIFO, random etc. The Client Cache Manager 207 is responsible for getting the application bits requested by the Client Streaming File System 212." and 53-57, "These requests lead to page faults in the operating system and the page faults are handled by the Client Streaming File System 212 that in turn asks the Client Cache Manager 207 for the appropriate bits.")*. Note that the cache stores the application bits that are provided to the Client Streaming File System when requested (installing the application). When the cache is fully occupied (reaches a threshold number), various cache algorithms may be used to replace existing portions of the cache so that other requested application bits can be provided to the Client Streaming File System when requested.

Therefore, for at least the reason set forth above, the rejections made under 35 U.S.C. § 103(a) with respect to Claims 27, 45, and 46 are proper and therefore, maintained.

Conclusion

16. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period

will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

17. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Qing Chen whose telephone number is 571-270-1071. The Examiner can normally be reached on Monday through Thursday from 7:30 AM to 4:00 PM. The Examiner can also be reached on alternate Fridays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Wei Zhen, can be reached on 571-272-3708. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571-272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

/Q. C./

Examiner, Art Unit 2191

/Wei Y Zhen/

Supervisory Patent Examiner, Art Unit 2191